

Minimum Makespan Multi-vehicle Dial-a-Ride*

Inge Li Gørtz[†]Viswanath Nagarajan[‡]R. Ravi[§]

Abstract

Dial-a-Ride problems consist of a set V of n vertices in a metric space (denoting travel time between vertices) and a set of m objects represented as source-destination pairs $\{(s_i, t_i)\}_{i=1}^m$, where each object requires to be moved from its source to destination vertex. In the *multi-vehicle Dial-a-Ride* problem, there are q vehicles each having capacity k and where each vehicle $j \in [q]$ has its own depot-vertex $r_j \in V$. A feasible schedule consists of a capacitated route for each vehicle (where vehicle j originates and ends at its depot r_j) that together move all objects from their sources to destinations. The objective is to find a feasible schedule that minimizes the maximum completion time (i.e. *makespan*) of vehicles, where the completion time of vehicle j is the time when it returns to its depot r_j at the end of its route. We study the *preemptive* version of multi-vehicle Dial-a-Ride, where an object may be left at intermediate vertices and transported by more than one vehicle, while being moved from source to destination.

Our main results are an $O(\log^3 n)$ -approximation algorithm for *preemptive multi-vehicle Dial-a-Ride*, and an improved $O(\log t)$ -approximation for its special case when there is no capacity constraint (here $t \leq n$ is the number of distinct depot-vertices). There is an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation known even for single vehicle capacitated Dial-a-Ride [17]. For uncapacitated multi-vehicle Dial-a-Ride, we show that there are instances where natural lower bounds (used in our algorithm) are $\tilde{\Omega}(\log t)$ factor away from the optimum.

We also consider the special class of metrics induced by graphs excluding any fixed minor (eg. planar metrics). In this case, we obtain improved guarantees of $O(\log^2 n)$ for capacitated multi-vehicle Dial-a-Ride, and $O(1)$ for the uncapacitated problem.

1 Introduction

The *multi-vehicle Dial-a-Ride* problem involves routing a set of m objects from their sources to respective destinations using a set of q vehicles starting at t distinct depot vertices in an n -vertex metric. Each vehicle has a *capacity* k which is the maximum number of objects it can carry at any time. Two versions arise based on whether or not the vehicle can use any vertex in the metric as a preemption (a.k.a. transshipment) point - we study the less-examined *preemptive version* in this paper. The objective in these problems is either the total completion time or the makespan (maximum completion time) over the q vehicles, and we study the more interesting *makespan* version of the problem. Thus this paper studies the preemptive, capacitated, minimum makespan, multi-vehicle Dial-a-Ride (mDaR) problem.

While the multiple qualifications may make the problem appear contrived, this is exactly the problem that models courier or mail delivery over a day from several city depots: preemption is cheap and useful for packages, trucks are capacitated and the makespan reflects the daily working time limit for each truck. Despite its ubiquity, this problem has not been as well studied as other Dial-a-Ride versions. One reason from the empirical side is the difficulty in handling the possibility of preemptions in a

*A preliminary version appeared in the Proceedings of the 17th Annual European Symposium on Algorithms (ESA) 2009.

[†]Technical University of Denmark. Sponsored in part by a grant from the Carlsberg Foundation.

[‡]IBM T.J. Watson Research Center.

[§]Tepper School of Business, Carnegie Mellon University. Supported in part by NSF grant CCF-0728841.

clean mathematical programming model. On the theoretical side which is the focus of this paper, the difficulty of using preemption in a meaningful way in an approximation algorithm persists. It is further compounded by the hardness of the makespan objective.

The requirement in preemptive Dial-a-Ride, that preemptions are allowed at all vertices, may seem unrealistic. In practice, a subset P of the vertex-set V might represent the vertices where preemption is permitted: the two extremes of this general problem are non-preemptive Dial-a-Ride ($P = \emptyset$) and preemptive Dial-a-Ride ($P = V$). However preemptive Dial-a-Ride is more generally applicable: specifically in situations where the preemption-points P form a *net* of the underlying metric (i.e. every vertex in V has a nearby preemption-point). In particular, approximation algorithms for preemptive Dial-a-Ride imply good approximations even in this general setting, wherein the precise guarantee depends on how well P covers V .

We also note that though our model allows any number of preemptions and preemptions at all vertices, our algorithms do not use this possibility to its full extent. Our algorithm for the capacitated case preempts each object at most once and our algorithm for the uncapacitated case only preempts objects at depot vertices (and at most $O(\log t)$ times).

The preemptive Dial-a-Ride problem has been considered earlier with a single vehicle, for which an $O(\log n)$ approximation [8] and an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation (for any constant $\epsilon > 0$) [17] are known. Note that the completion time and makespan objectives coincide for this case.

Moving to multiple vehicles, the total completion time objective admits a straightforward $O(\log n)$ approximation along the lines of the single vehicle problem [8]: Using the FRT tree embedding [15], one can reduce to tree-metrics at the loss of an expected $O(\log n)$ factor, and there is a simple constant factor approximation for this problem on trees. The maximum completion time or makespan objective, which we consider in this paper turns out to be considerably harder. Due to non-linearity of the makespan objective, the above reduction to tree-metrics does not hold. Furthermore, the makespan objective does not appear easy to solve even on trees.

Unlike in the single-vehicle case, note that an object in multi-vehicle Dial-a-Ride may be transported by several vehicles one after the other. Hence it is important for the vehicle routes to be coordinated so that the objects trace valid paths from respective sources to destinations. For example, a vehicle may have to wait at a vertex for other vehicles carrying common objects to arrive. The multi-vehicle Dial-a-Ride problem captures aspects of both machine scheduling and network design problems.

1.1 Results, Techniques and Paper Outline

Uncapacitated mDaR. We first consider the special case of multi-vehicle Dial-a-Ride where the vehicles have no capacity constraints (i.e. $k \geq m$). This problem is interesting in itself, and serves as a good starting point before we present the algorithm for the general case. The uncapacitated mDaR problem itself highlights differences from the single vehicle case: For example, in single vehicle Dial-a-Ride, preemption plays no role in the absence of capacity constraints; whereas in the multi-vehicle case, an optimal non-preemptive schedule may take $\Omega(\sqrt{q})$ longer than the optimal preemptive schedule (see Section 2). Our first main result is the following:

Theorem 1 *There is an $O(\log t)$ -approximation algorithm for uncapacitated preemptive mDaR. Additionally, the schedule preempts objects only at depot vertices.*

The above algorithm has two main steps: the first one (in Subsection 2.1) reduces the instance at a constant factor loss in the performance guarantee to one in which all demands are between depots (a “depot-demand” instance). In the second step (Subsection 2.2), we use a *sparse spanner* on the demand graph to construct routes for moving objects across depots.

We also construct instances of uncapacitated mDaR where the optimal value is $\Omega(\log t / \log \log t)$ times all our lower bounds for this problem (Subsection 2.3). This suggests that stronger lower bounds are needed to obtain a better approximation ratio than what our approach provides.

We then consider the special class of metrics induced by graphs excluding some fixed minor (such as planar or bounded-genus graphs), and obtain the following improved guarantee in Subsection 2.4.

Theorem 2 *There is an $O(1)$ -approximation algorithm for uncapacitated preemptive mDaR on metrics induced by graphs that exclude any fixed minor.*

Again the resulting schedule only preempts objects at depot-vertices. Furthermore, each object is preempted at most thrice; whereas the algorithm in Theorem 1 might preempt each object $O(\log t)$ times. The algorithm in Theorem 2 has the same high-level approach outlined for Theorem 1: the difference is in the second step, where we use a stronger notion of *sparse covers* in such metrics (which follows from the KPR decomposition theorem [22]), to construct routes for moving objects across depots.

Capacitated mDaR. In Section 3, we study the capacitated multi-vehicle Dial-a-Ride problem, and obtain our second main result. Recall that there is an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation for even single vehicle Dial-a-Ride [17].

Theorem 3 *There is a randomized $O(\log^3 n)$ -approximation algorithm for preemptive mDaR. Additionally, the schedule preempts each object at most once.*

This algorithm is considerably more complex than the one for the uncapacitated special case, and is the main technical contribution of this paper. It has four key steps: (1) We *preprocess* the input so that demand points that are sufficiently far away from each other can be essentially decomposed into separate instances for the algorithm to handle independently. (2) We then solve a single-vehicle instance of the problem that obeys some additional *bounded-delay* property (Theorem 9) that we prove; This property combines ideas from algorithms for *light approximate shortest path trees* [21] and *capacitated vehicle routing* [19]. The bounded-delay property is useful in *randomly partitioning* the single vehicle solution among the q vehicles available to share this load. This random partitioning scheme is reminiscent of the work of Hochbaum-Maass [20], Baker [3] and Klein-Plotkin-Rao [22], in trying to average out the effect of the cutting in the objective function. (3) The partitioned segments of the single vehicle tour are assigned to the available vehicles; However, to check if this assignment is feasible we solve a matching problem that identifies cases when this load assignment must be *rebalanced*. This is perhaps the most interesting step in the algorithm since it identifies stronger lower bounds for subproblems where the current load assignment is not balanced. (4) We finish up by *recursing* on the load rebalanced subproblem; An interesting feature of the recursion is that the fraction of demands that are processed recursively is not a fixed value (as is more common in such recursive algorithms) but is a carefully chosen function of the number of vehicles on which these demands have to be served.

We prove the new bounded-delay property of single-vehicle Dial-a-Ride in Subsection 3.1. Then we present the algorithm for Theorem 3 in Subsection 3.2 and establish an $O(\log^2 m \log n)$ -approximation bound. Using some additional preprocessing, we show in Subsection 3.3 how to remove the dependence on m (number of objects) to obtain the final $O(\log^3 n)$ approximation ratio.

When the underlying metric is induced by graphs excluding a fixed minor, we can establish a stronger bounded-delay property in step (2) of the above framework. The main idea here is the construction of *well-separated covers* in such metrics, which we show can be obtained using the KPR decomposition [22]. This leads to the following improved guarantee, proved in Subsection 3.4.

Theorem 4 *There is an $O(\log^2 n)$ -approximation algorithm for preemptive mDaR on metrics induced by graphs excluding any fixed minor.*

1.2 Related Work

Dial-a-Ride problems form an interesting subclass of Vehicle Routing Problems that are well studied in the operations research literature. Pape et al. [13] provide a classification of Dial-a-Ride problems using a notation similar to that for scheduling and queuing problems: preemption is one aspect in this

classification. Savelsberg and Sol [27] and Cordeau and Laporte [12] survey several variants of non-preemptive Dial-a-Ride problems that have been studied in the literature. Most Dial-a-Ride problems arising in practice involve making routing decisions for multiple vehicles.

Dial-a-Ride problems with transshipment (the preemptive version) have been studied in [24, 25, 26]. These papers consider a more general model where preemption is allowed only at a specified subset of vertices. Our model (and that of [8]) is the special case when every vertex can serve as a preemption point. It is clear that preemption only reduces the cost of serving demands: Nakao and Nagamochi [26] studied the maximum decrease in the optimal cost upon introducing one preemption point. [24, 25] also model time-windows on the demands, and study heuristics and a column-generation based approach; they also describe applications (eg. courier service) that allow for preemptions.

For single vehicle Dial-a-Ride, the best known approximation guarantee for the preemptive version is $O(\log n)$ (Charikar and Raghavachari [8]), and an $\Omega(\log^{1/4-\epsilon} n)$ hardness of approximation (for any constant $\epsilon > 0$) is shown in Gørtz [17]. The non-preemptive version appears much harder and the best known approximation ratio is $\min\{\sqrt{k} \log n, \sqrt{n} \log^2 n\}$ (Charikar and Raghavachari [8], Gupta et al. [18]); however to the best of our knowledge, APX-hardness is the best lower bound. There are known instances of single vehicle Dial-a-Ride where the ratio between optimal non-preemptive and preemptive tours is $\Omega(\sqrt{n})$ in general metrics [8], and $\tilde{\Omega}(n^{1/8})$ in the Euclidean plane [18]. A 1.8-approximation is known for the $k = 1$ special case of single vehicle Dial-a-Ride (a.k.a. *stacker-crane* problem) [16].

Charikar et al. [7] studied the related k -delivery TSP problem, which involves transporting a number of *identical objects* from supply to demand vertices, using a single capacity k vehicle. The key difference from Dial-a-Ride is that an object can be moved from any supply vertex to any demand vertex. [7] gave an approximation algorithm for k -delivery TSP that outputs a non-preemptive tour of length at most five times an optimal preemptive tour. They also showed that for any k -delivery TSP instance, the optimal non-preemptive tour is at most four times the optimal preemptive tour.

The *truck and trailer routing problem* [5, 28] is another problem where preemption plays a crucial role. Here a number of capacitated trucks and trailers are used to deliver all objects. Some customers are only accessible without the trailer. The trailers can be parked at any point accessible with a trailer and it is possible to shift demand loads between the truck and the trailer at the parking places. The papers [5, 28] present heuristics for this problem.

Single vehicle preemptive Dial-a-Ride is closely related to the uniform *buy-at-bulk* problem (with cost function $\lceil \frac{x}{k} \rceil$ where k is the vehicle capacity). Such a connection was formally used in Gørtz [17] to establish the hardness of approximation for the single vehicle problem. Approximation algorithms for several buy-at-bulk variants have been studied recently, eg. non-uniform buy-at-bulk [6, 9], buy-at-bulk with node-costs [10] and buy-at-bulk with protection [2]; poly-logarithmic approximation guarantees are known for all these problems. However the techniques required to solve the *multi-vehicle* Dial-a-Ride problem appear quite different from these buy-at-bulk results.

The uncapacitated mDaR problem generalizes the *nurse-station-location* problem that was studied in Even et al. [14] (where a 4-approximation algorithm was given). In fact we also use this algorithm as a subroutine for uncapacitated mDaR. Nurse-station-location is the special case of uncapacitated mDaR when each source-destination pair coincides on a single vertex. In this paper, we handle not only the case with arbitrary pairs (uncapacitated mDaR), but also the more general problem (capacitated mDaR) with finite capacity restriction.

1.3 Problem Definition and Preliminaries

We represent a finite metric as (V, d) where V is the set of vertices and d is a symmetric distance function satisfying the triangle inequality. For subsets $A, B \subseteq V$ we denote by $d(A, B)$ the minimum distance between a vertex in A and another in B , so $d(A, B) = \min\{d(u, v) \mid u \in A, v \in B\}$. For a subset $E \subseteq \binom{V}{2}$ of edges, $d(E) := \sum_{e \in E} d_e$ denotes the total length of edges in E .

The *multi-vehicle Dial-a-Ride problem* (mDaR) consists of an n -vertex metric (V, d) , m objects specified as source-destination pairs $\{s_i, t_i\}_{i=1}^m$, q vehicles having respective depot-vertices $\{r_j\}_{j=1}^q$,

and a common vehicle capacity k . A feasible schedule is a set of q routes, one for each vehicle (where the route for vehicle $j \in [q]$ starts and ends at r_j), such that no vehicle carries more than k objects at any time and each object is moved from its source to destination. The completion time C_j of any vehicle $j \in [q]$ is the time when vehicle j returns to its depot r_j at the end of its route (the schedule is assumed to start at time 0). The objective in mDaR is to minimize the makespan, i.e., $\min \max_{j \in [q]} C_j$. We denote by $S := \{s_i \mid i \in [m]\}$ the set of sources, $T := \{t_i \mid i \in [m]\}$ the set of destinations, $R := \{r_j \mid j \in [q]\}$ the set of distinct depot-vertices, and $t := |R|$ the number of distinct depots. In this paper, we only consider the *preemptive* version, where objects may be left at intermediate vertices while being moved from source to destination.

Single vehicle Dial-a-Ride. The following are lower bounds for the single vehicle problem: the minimum length TSP tour on the depot and all source/destination vertices (*Steiner* lower bound), and $\frac{\sum_{i=1}^m d(s_i, t_i)}{k}$ (*flow* lower bound). Charikar and Raghavachari [8] gave an $O(\log n)$ -approximation algorithm for this problem based on the above lower bounds. A feasible solution to preemptive Dial-a-Ride is said to be *1-preemptive* if every object is preempted at most once while being moved from its source to destination. Gupta et al. [18] showed that the single vehicle preemptive Dial-a-Ride problem always has a 1-preemptive tour of length $O(\log^2 n)$ times the Steiner and flow lower-bounds.

Lower bounds for mDaR. The quantity $\frac{\sum_{i=1}^m d(s_i, t_i)}{qk}$ is a lower bound similar to the flow bound for single vehicle Dial-a-Ride. Analogous to the Steiner lower bound above, is the optimal value of an induced *nurse-station-location* instance. In the nurse-station-location problem [14], we are given a metric (V, d) , a set \mathcal{T} of terminals and a multi-set $\{r_j\}_{j=1}^q$ of depot-vertices; the goal is to find a collection $\{F_j\}_{j=1}^q$ of trees that collectively contain all terminals \mathcal{T} such that each tree F_j is rooted at vertex r_j and $\max_{j=1}^q d(F_j)$ is minimized. Even et al. [14] gave a 4-approximation algorithm for this problem. The optimal value of the nurse-station-location instance with depots $\{r_j\}_{j=1}^q$ (depots of vehicles in mDaR) and terminals $\mathcal{T} = S \cup T$ is a lower bound for mDaR. The following are some lower bounds implied by nurse-station-location: (a) $1/q$ times the minimum length forest that connects every vertex in $S \cup T$ to some depot vertex, (b) $\max_{i \in [m]} d(R, s_i)$, and (c) $\max_{i \in [m]} d(R, t_i)$. Finally, it is easy to see that $\max_{i \in [m]} d(s_i, t_i)$ is also a lower bound for mDaR.

We note that our approximation bounds for uncapacitated mDaR are relative to the above lower bounds. However the algorithm for capacitated mDaR relies additionally on stronger lower bounds derived from suitable subproblems.

2 Uncapacitated Multi-Vehicle Dial-a-Ride

In this section we study the uncapacitated special case of mDaR, where vehicles have no capacity constraints (i.e. capacity $k \geq m$). We give an algorithm that achieves an $O(\log t)$ approximation ratio for this problem (recall $t \leq n$ is the number of distinct depots). Unlike in the single vehicle case, preemptive and non-preemptive versions of mDaR are very different even without capacity constraints.

Preemption gap in Uncapacitated mDaR. Consider an instance of uncapacitated mDaR where the metric is induced by an unweighted star with q leaves (where q is number of vehicles), all q vehicles have the center vertex as depot, and there is a demand between every pair of leaf-vertices. A preemptive schedule having makespan 4 is as follows: each vehicle $j \in [q]$ visits leaf j and brings all demands with source j to the root, then each vehicle j visits its corresponding leaf again, this time delivering all demands with destination j . On the other hand, in any non-preemptive schedule, one of the q vehicles completely serves at least $q - 1$ demands (since there are $q(q - 1)$ demands in all). The minimum length of any tour containing the end points of q demands is $\Omega(\sqrt{q})$, which is also a lower bound on the optimal non-preemptive makespan. Thus there is an $\Omega(\sqrt{q})$ factor gap between optimal preemptive and

non-preemptive tours. This is in contrast to the *uncapacitated single vehicle* case, where it is easy to see that the optimal preemptive and non-preemptive tours coincide.

The algorithm for uncapacitated mDaR proceeds in two stages. Given any instance, it is first reduced (at the loss of a constant factor) to a depot-demand instance, where all demands are between depot vertices (Subsection 2.1). This reduction uses the nurse-station-location algorithm from Even et al. [14]. Then the depot-demand instance is solved using an $O(\log t)$ -approximation algorithm (Subsection 2.2); this is the key step in the algorithm, and is based on a sparse-spanner on the demand graph.

2.1 Reduction to Depot-demand Instances

We define *depot-demand instances* as those instances of uncapacitated mDaR where all demands are between depot vertices. Given any instance \mathcal{I} of uncapacitated mDaR, the algorithm **UncapMulti** (Figure 1) reduces \mathcal{I} to a depot-demand instance. We now argue that the reduction in **UncapMulti** only loses a constant approximation factor. Let B denote the optimal makespan of instance \mathcal{I} . Since the optimal value of the nurse-station-location instance solved in the first step of **UncapMulti** is a lower bound for \mathcal{I} , we have $\max_{j=1}^q d(F_j) \leq 4B$.

Input: instance \mathcal{I} of uncapacitated mDaR.

- Solve the nurse-station-location instance with depots $\{r_j\}_{j=1}^q$ and all sources/destinations $S \cup T$ as terminals, using the 4-approximation algorithm [14]. Let $\{F_j\}_{j=1}^q$ be the resulting trees covering $S \cup T$ such that each tree F_j is rooted at depot r_j .
- Define a depot-demand instance \mathcal{J} of uncapacitated mDaR on the same metric and set of vehicles, where the demands are $\{(r_j, r_l) \mid s_i \in F_j \text{ \& } t_i \in F_l, 1 \leq i \leq m\}$. For any object $i \in [m]$ let the *source depot* be the depot r_j for which $s_i \in F_j$ and the *destination depot* be the depot r_l for which $t_i \in F_l$.
- Output the following schedule for \mathcal{I} :
 1. Each vehicle $j \in [q]$ traverses tree F_j by an Euler tour, picks up all objects from sources in F_j and brings them to their source-depot r_j .
 2. Vehicles implement a schedule for *depot-demand instance* \mathcal{J} , and all objects are moved from their source-depot to destination-depot (using the algorithm in Section 2.2).
 3. Each vehicle $j \in [q]$ traverses tree F_j by an Euler tour, picks up all objects having destination-depot r_j and brings them to their destinations in F_j .

Figure 1: Algorithm **UncapMulti** for uncapacitated mDaR.

Claim 5 *The optimal makespan for the depot-demand instance \mathcal{J} is at most $17B$.*

Proof: Consider a feasible schedule for \mathcal{J} involving three rounds: (1) each vehicle traverses (by means of an Euler tour) its corresponding tree in $\{F_j\}_{j=1}^q$ and moves each object i from its source-depot (the source in instance \mathcal{J}) to s_i (source in original instance \mathcal{I}); (2) each vehicle follows the optimal schedule for \mathcal{I} and moves each object i from s_i to t_i (destination in \mathcal{I}); (3) each vehicle traverses its corresponding tree in $\{F_j\}_{j=1}^q$ and moves each object i from t_i to its destination-depot (the destination in \mathcal{J}). Clearly this is a feasible schedule for \mathcal{J} . From the observation on the nurse-station-location instance, the time taken in each of the first and third rounds is at most $8B$. Furthermore, the time taken in the second round is the optimal makespan of \mathcal{I} which is B . This proves the claim. ■

Assuming a feasible schedule for \mathcal{J} , it is clear that the schedule returned by **UncapMulti** is feasible for the original instance \mathcal{I} . The first and third rounds in \mathcal{I} 's schedule require at most $8B$ time each.

Thus an approximation ratio α for depot-demand instances implies an approximation ratio of $17\alpha + 8$ for general instances. In the next subsection, we show an $O(\log t)$ -approximation algorithm for depot-demand instances (here t is the number of depots), which implies Theorem 1.

2.2 Algorithm for Depot-demand Instances

Let \mathcal{J} be any depot-demand instance: note that the instance defined in the second step of **UncapMulti** is of this form. It suffices to restrict the algorithm to the induced metric (R, d) on only depot vertices, and use only one vehicle at each depot in R . Consider an undirected graph H consisting of vertex set R and edges corresponding to demands: there is an edge between vertices r and s iff there is an object going from either r to s or s to r . Note that the metric length of any edge in H is at most the optimal makespan \tilde{B} of instance \mathcal{J} . In the schedule produced by our algorithm, vehicles will only use edges of H . Thus in order to obtain an $O(\log t)$ approximation, it suffices to show that each vehicle only traverses $O(\log t)$ edges. Based on this, we further reduce \mathcal{J} to the following instance \mathcal{H} of uncapacitated mDaR: the underlying metric is shortest paths in graph H (on vertices R), with one vehicle at each R -vertex, and for every edge $(u, v) \in H$ there is a demand from u to v and one from v to u . Clearly any schedule for \mathcal{H} having makespan β implies one for \mathcal{J} of makespan $\beta \cdot \tilde{B}$. The next lemma implies an $O(\log |R|)$ approximation for depot-demand instances.

Lemma 6 *There exists a poly-time computable schedule for \mathcal{H} with makespan $O(\log t)$, where $t = |R|$.*

Proof: Let $\alpha = \lceil \lg t \rceil + 1$. We first construct a *sparse spanner* A of H as follows: consider edges of H in an arbitrary order, and add an edge $(u, v) \in H$ to A iff the shortest path between u and v using current edges of A is more than 2α . It is clear from this construction that the girth of A (length of its shortest cycle) is at least 2α , and that for every edge $(u, v) \in H$, the shortest path between u and v in A is at most 2α .

We now assign each edge of A to one of its end-points such that each vertex is assigned at most two edges. Repeatedly pick any vertex v of degree at most two in A , assign its adjacent edges to v , and remove these edges and v from A . We claim that at the end of this procedure (when no vertex has degree at most 2), all edges of A would have been removed (i.e. assigned to some vertex). Suppose for a contradiction that this is not the case. Let $\tilde{A} \neq \emptyset$ be the remaining graph; note that $\tilde{A} \subseteq A$, so girth of \tilde{A} is at least 2α . Every vertex in \tilde{A} has degree at least 3, and there is at least one such vertex w . Consider performing a breadth-first search in \tilde{A} from w . Since the girth of \tilde{A} is at least 2α , the first α levels of the breadth-first search is a tree. Furthermore every vertex has degree at least 3, so each vertex in the first $\alpha - 1$ levels has at least 2 children. This implies that \tilde{A} has at least $1 + 2^{\alpha-1} > t$ vertices, which is a contradiction! For each vertex $v \in R$, let A_v denote the edges of A assigned to v by the above procedure; we argued that $\cup_{v \in R} A_v = A$, and $|A_v| \leq 2$ for all $v \in R$.

The schedule for \mathcal{H} involves 2α rounds as follows. In each round, every vehicle $v \in R$ traverses the edges in A_v (in both directions) and returns to v . Since $|A_v| \leq 2$ for all vertices v , each round takes 4 units of time; so the makespan of this schedule is $8\alpha = O(\log t)$. The route followed by each object in this schedule is the shortest path from its source to destination in spanner A ; note that the length of any such path is at most 2α . To see that this is indeed feasible, observe that every edge of A is traversed by some vehicle in each round. Hence in each round, every object traverses one edge along its shortest path (unless it is already at its destination). Thus after 2α rounds, all objects are at their destinations. ■

Note that the final algorithm for uncapacitated mDaR preempts each object at most $O(\log t)$ times, and only at depot-vertices. This completes the proof of Theorem 1.

2.3 Tight Example for Uncapacitated mDaR Lower Bounds

We note that known lower bounds for uncapacitated mDaR are insufficient to obtain a sub-logarithmic approximation guarantee. The lower bounds we used in our algorithm are the following: $\max_{i \in [m]} d(s_i, t_i)$,

and the optimal value of a nurse-station-location instance with depots $\{r_j\}_{j=1}^q$ and terminals $S \cup T$. We are not aware of any lower bounds stronger than these two bounds.

We show an instance \mathcal{G} of uncapacitated mDaR where the optimal makespan is a factor $\Omega(\frac{\log t}{\log \log t})$ larger than both the above lower bounds. In fact, the instance we construct is a depot-demand instance that has the same special structure as instance \mathcal{H} in Lemma 6. I.e. the demand graph is same as the graph inducing distances. Take $G = (V, E)$ to be a t -vertex regular graph of degree $\sim \log t$ and girth $g \sim \log t / \log \log t$ (there exist such graphs, eg. Lazebnik et al. [23]). Instance \mathcal{G} is defined on a metric on vertices V with distances being shortest paths in graph G . For every edge $(u, v) \in E$ of graph G , there is an object with source u and destination v (the direction is arbitrary). There is one vehicle located at every vertex of V ; so number of vehicles $q = t$.

Observe that both our lower bounds are $O(1)$: the optimal value of the nurse-station-location instance is 0, and maximum source-destination distance is 1. However as we show below, the optimal makespan for this instance is at least $g - 1 = \Omega(\log t / \log \log t)$. Suppose (for contradiction) that there is a feasible schedule for \mathcal{G} with makespan $M < g - 1$. A demand $(u, v) \in E$ is said to be *completely served* by a vehicle j iff the route of vehicle j visits both vertices u and v . The number of distinct vertices visited by any single vehicle is at most $M < g$; so the number of demands that are completely served by a single vehicle is at most $M - 1$ (otherwise these demand edges would induce a cycle smaller than the girth g). Hence the number of demands that are completely served by some vehicle is at most $t \cdot g < |E|$. Let $(u, v) \in E$ be a demand that is *not* completely served by any vehicle, i.e. there is no vehicle that visits both u and v . Since we have a feasible schedule of makespan M , the path π followed by demand (u, v) from u to v (or vice versa) in the schedule has length at most M . The path π can not be the direct edge (u, v) since demand (u, v) is not completely served by any vehicle. So path π together with edge (u, v) is a cycle of length at most $M + 1 < g$ in graph G , contradicting girth of G .

2.4 Improved Algorithm for Metrics Excluding a Fixed Minor

We now prove Theorem 2 that gives a constant approximation algorithm for uncapacitated mDaR on metrics induced by K_r -minor free graphs (for any fixed r). This improvement comes from using the existence of good ‘sparse covers’ in such metrics, as opposed to the spanner based construction in Lemma 6. This guarantee is again relative to the above mentioned lower bounds.

Consider an instance of uncapacitated mDaR on metric (V, d) that is induced by an edge-weighted graph $G = (V, E)$ containing no K_r -minor (for some fixed $r \geq 1$). We start with some definitions [4]. A *cluster* is any subset of vertices. For any $\gamma > 0$ and vertex $v \in V$, $N(v, \gamma) := \{u \in V \mid d(u, v) \leq \gamma\}$ denotes the set of vertices within distance γ from v . As observed in Busch et al. [4] and Abraham et al. [1], the partitioning scheme of Klein et al. [22] implies the following result.

Theorem 7 ([22]) *Given K_r -minor free graph $G = (V, E, w)$ and value $\gamma > 0$, there is an algorithm that computes a set $Z = \{C_1, \dots, C_l\}$ of clusters satisfying:*

1. *The diameter of each cluster is at most $O(r^2) \cdot \gamma$, i.e. $\max_{u, v \in C_i} d(u, v) \leq O(r^2) \cdot \gamma$ for all $i \in [l]$.*
2. *For every $v \in V$, there is some cluster $C_i \in Z$ such that $N(v, \gamma) \subseteq C_i$.*
3. *For every $v \in V$, the number of clusters in Z that contain v is at most $O(2^r)$.*

The set of clusters Z found above is called a *sparse cover* of G .

Proof: [Theorem 2] The reduction in Section 2 implies that it suffices to consider depot-demand instances: An $O(1)$ approximation for such instances implies an $O(1)$ approximation for general instances. Let \mathcal{J} be any depot-demand instance on metric (V, d) induced by K_r -minor free graph $G = (V, E)$, with a set $R \subseteq V$ of depot-vertices (each containing a vehicle), and where all demands $\{s_i, t_i\}_{i=1}^m$ are between vertices of R . The algorithm is described in Figure 2.

Input: Depot-demand instance \mathcal{J} on metric $G = (V, E, w)$, depot-vertices $R \subseteq V$, demands $\{s_i, t_i\}_{i=1}^m$.

1. Let $\gamma = \max_{i \in [m]} d(s_i, t_i)$ be the maximum source-destination distance.
2. Compute a sparse cover $Z = \{C_j\}_{j=1}^l$ given in Theorem 7 for parameter γ .
3. For each cluster $C_j \in Z$, choose an arbitrary vertex $c_j \in C_j$ as its *center*.
4. For each demand $i \in [m]$, let $\pi(i) \in [l]$ be such that $s_i, t_i \in C_{\pi(i)}$.
5. Output the following schedule for \mathcal{J} :
 - (a) Each vehicle $r \in R$ visits the centers of all clusters containing r , and returns to r . Vehicle r carries all the objects $\{i \in [m] \mid s_i = r\}$ having source r , and drops each object i at center $c_{\pi(i)}$.
 - (b) Each vehicle $r \in R$ again visits the centers of all clusters containing r . Vehicle r brings all the objects $\{i \in [m] \mid t_i = r\}$ having destination r : each object i is picked up from center $c_{\pi(i)}$.

Output: An $O(1)$ -approximate minimum makespan schedule for \mathcal{J} .

Figure 2: Algorithm for uncapacitated mDaR on K_r -minor free graphs.

Note that γ is a lower bound on the optimal makespan of \mathcal{J} . We claim that the makespan of the above schedule is at most $O(r^2 2^r) \cdot \gamma$. Observe that each depot is contained in at most $O(2^r)$ clusters, and the distance from any depot to the center of any cluster containing it is at most $O(r^2) \cdot \gamma$. Hence the time taken by each vehicle in either of the two rounds (Steps (5a)-(5b)) is at most $O(r^2 2^r) \cdot \gamma$. Since r is a fixed constant, the final makespan is $O(1) \cdot \gamma$.

We now argue the feasibility of the above schedule. Step (4) is well-defined: for all $i \in [m]$, we have $s_i, t_i \in N(s_i, \gamma)$ and there is some $j \in [l]$ with $N(s_i, \gamma) \subseteq C_j$ (i.e. we can set $\pi(i) = j$). It is now easy to see that each object $i \in [m]$ traces the following route in the final schedule: $s_i \rightsquigarrow c_{\pi(i)} \rightsquigarrow t_i$. ■

Combining this algorithm for depot-demand instances with the reduction in Subsection 2.1, we obtain an $O(r^2 2^r)$ -approximation algorithm for uncapacitated mDaR on K_r -minor free metrics. When r is a fixed constant this is a constant approximation, and we obtain Theorem 2. Note that each object in the resulting schedule is preempted at most thrice, and only at depot-vertices.

3 Capacitated Multi-Vehicle Dial-a-Ride

In this section we prove our main result: an $O(\log^3 n)$ -approximation algorithm for the mDaR problem (with capacities). We first obtain a new structure theorem on single-vehicle Dial-a-Ride tours (Subsection 3.1) that preempts each object at most once, and where the total time spent by objects in the vehicle is bounded. Obtaining such a single vehicle tour is crucial in our algorithm for capacitated mDaR, which appears in Section 3.2. There we prove a weaker approximation bound of $O(\log^2 m \log n)$; in Subsection 3.3 we show how to remove the dependence on m to obtain an $O(\log^3 n)$ -approximation algorithm even for “weighted mDaR” (i.e. Theorem 3). In Subsection 3.4, we establish a stronger structure theorem (analogous to the one in Subsection 3.1) for metrics excluding any fixed minor; this immediately leads to an improved $O(\log^2 n)$ -approximation ratio for mDaR on such metrics (i.e. Theorem 4).

3.1 Capacitated Vehicle Routing with Bounded Delay

Before we present the structural result on Dial-a-Ride tours, we consider the classic *capacitated vehicle routing problem* [19] with an additional constraint on object ‘delays’. The capacitated vehicle routing problem (CVRP) is a special case of single vehicle Dial-a-Ride when all objects have the same source (or

equivalently, same destination). Formally, we are given a metric (V, d) , specified depot-vertex $r \in V$, and m objects all having source r and respective destinations $\{t_i\}_{i \in [m]}$. The goal is to compute a minimum length *non-preemptive* tour of a capacity k vehicle originating at r that moves all objects from r to their destinations. In *CVRP with bounded delay*, we are additionally given a *delay parameter* $\beta > 1$, and the goal is to find a minimum length capacitated non-preemptive tour serving all objects such that the time spent by each object $i \in [m]$ in the vehicle is at most $\beta \cdot d(r, t_i)$. Well-known lower bounds for the CVRP [19] are as follows: minimum length TSP tour on $\{r\} \cup \{t_i \mid i \in [m]\}$ (Steiner lower bound), and $\frac{2}{k} \sum_{i=1}^m d(r, t_i)$ (flow lower bound). These lower bounds also hold for the (less constrained) preemptive version of CVRP.

Theorem 8 *There is a $(2.5 + \frac{3}{\beta-1})$ -approximation algorithm for CVRP with bounded delay, where $\beta > 1$ is the delay parameter. This guarantee is relative to the Steiner and flow lower bounds.*

Proof: Our algorithm is basically a combination of the algorithms for *light approximate shortest path trees* [21], and capacitated vehicle routing [19]. Let LB denote the maximum of the Steiner and flow lower bounds. The minimum TSP tour on the destinations plus r is the Steiner lower bound. The first step is to compute an approximately minimum TSP tour C : Christofides' algorithm [11] gives a 1.5-approximation, so $d(C) \leq 1.5 \cdot \text{LB}$. Number the vertices in the order in which they appear in C , starting with r being 0. Using the procedure in Khuller et al. [21], we obtain a set of edges $\{(0, v_1), \dots, (0, v_t)\}$ with $0 < v_1 < v_2 < \dots < v_t < |V|$ having the following properties (below $v_0 = 0$).

1. For $1 \leq p \leq t$, for any vertex u with $v_{p-1} \leq u < v_p$, the length of edge $(0, v_{p-1})$ plus the path along C from v_{p-1} to u is at most $\beta \cdot d(0, u)$.
2. $\sum_{p=1}^t d(0, v_p) \leq \frac{1}{\beta-1} \cdot d(C)$.

For each $1 \leq p \leq t$, define tour C_p which starts at r , goes to v_{p-1} , traverses C until v_p , then returns to r . Assign vertices $\{v_{p-1}, \dots, v_p - 1\}$ (and all demands contained in them) to C_p . Also define tour C_{t+1} which starts at r , goes to v_t , and traverses C until r ; and assign all remaining demands to C_{t+1} . It is clear that C_p (for $1 \leq p \leq t+1$) visits each vertex assigned to it within β times the shortest path from r (using property 1 above). Also, the total length $\sum_{p=1}^{t+1} d(C_p) = d(C) + 2 \sum_{p=1}^t d(0, v_p) \leq (1 + \frac{2}{\beta-1})d(C)$, by property 2.

For each C_p , we service the set D_p of demands assigned to it separately. Index the demands in D_p in the order in which they appear on C_p (breaking ties arbitrarily). Consider a capacitated tour which serves these demands in groups of at most k each, and returns to r after serving each group. The groups are defined as follows: starting at index 1, each group contains the next k contiguous demands (until all of D_p is assigned to groups). By rotating the indexing of demands, there are k different groupings of D_p that can be obtained: each of which corresponds to a capacitated tour. As argued in [19] (and is easy to see), the average length of these k tours is at most $d(C_p) + 2 \sum_{z \in D_p} \frac{d(r, z)}{k}$. So the minimum length tour γ_p among these satisfies $d(\gamma_p) \leq d(C_p) + 2 \sum_{z \in D_p} \frac{d(r, z)}{k}$.

The final solution γ is the concatenation of tours $\gamma_1, \dots, \gamma_{t+1}$. Note that the time spent in the vehicle by any demand i is at most $\beta \cdot d(r, t_i)$. The length of tour γ is at most $\sum_{p=1}^{t+1} d(C_p) + 2 \sum_{z \in D} \frac{d(r, z)}{k}$, where D is the set of all demands. Note that $2 \sum_{z \in D} \frac{d(r, z)}{k}$ is the flow lower bound for this CVRP instance, so it is at most LB. Hence we obtain the following.

$$d(\gamma) \leq (1 + \frac{2}{\beta-1})d(C) + \text{LB} \leq (1 + \frac{2}{\beta-1})\frac{3}{2}\text{LB} + \text{LB} = (2.5 + \frac{3}{\beta-1})\text{LB}$$

Clearly, solution γ satisfies the desired conditions in the theorem. ■

We now consider the *single vehicle* preemptive Dial-a-Ride problem given by a metric (V, d) , a set D of demand-pairs, and a vehicle of capacity k . Given Theorem 8, the following structural result is a simple extension of Theorem 16 from Gupta et al. [18]. We present the proof for completeness.

Theorem 9 *There is a randomized poly-time computable 1-preemptive tour τ servicing D that satisfies the following conditions (where LB is the average of the Steiner and flow lower bounds):*

1. **Total length:** $d(\tau) \leq O(\log^2 n) \cdot \text{LB}$.
2. **Bounded delay:** $\sum_{i \in D} T_i \leq O(\log n) \sum_{i \in D} d(s_i, t_i)$ where T_i is the total time spent by object $i \in D$ in the vehicle under the schedule given by τ .

Proof: We follow the algorithm of Gupta et al. [18] that obtains a single vehicle 1-preemptive tour within $O(\log^2 n)$ factor of the Steiner and flow lower bounds. In addition, we will also ensure the bounded-delay property. Using the results on probabilistic tree embedding [15], we may assume that the given metric is a *hierarchically well-separated tree* \mathcal{T} with distance function κ . This only increases the expected value of the lower bound by a factor of $O(\log n)$; i.e. $E[\widetilde{\text{LB}}] = O(\log n) \text{LB}$ where $\widetilde{\text{LB}}$ (resp. LB) equals the average of the Steiner and flow lower bounds on metric κ (resp. d). Furthermore, as discussed in [18], we may assume that the metric κ is induced by a tree \mathcal{T} on the original vertex set V having $l = O(\log n)$ levels.

We now partition the demands in \mathcal{T} into l sets with D_p (for $p = 1, \dots, l$) consisting of all demands $i \in D$ having their nearest common ancestor (nca) in level p (i.e. the nca of s_i and t_i is a vertex in level p). We service each D_p separately using a tour of length $O(\widetilde{\text{LB}})$. Finally we concatenate the tours for each level p , to obtain the theorem.

Servicing D_p . For each vertex v at level p in \mathcal{T} , let L_v denote the demands in D_p that have v as their nca. Let $\text{LB}(v)$ denote the average of the Steiner and flow lower-bounds (in metric κ) for the single vehicle problem with depot v and demands L_v . Clearly the flow lower bounds are disjoint for different vertices v . Also, the subtrees under any two different level p vertices are disjoint, so the Steiner lower bounds are disjoint for different v . Thus $\sum_v \text{LB}(v) \leq \widetilde{\text{LB}}$. We now show how each L_v is served separately.

Servicing L_v . Consider the following two instances of the CVRP problem in metric κ : \mathcal{I}_{src} with all sources in L_v and common destination v , and \mathcal{I}_{dest} with common source v and all destinations in L_v . Observe that the Steiner and flow lower bounds for each of \mathcal{I}_{src} and \mathcal{I}_{dest} are all at most 2LB_v .

Consider instance \mathcal{I}_{src} . Setting delay parameter $\beta = 2$ in Theorem 8, we obtain a *non-preemptive* tour σ_v that moves all objects in L_v from their sources to vertex v , such that (1) the length of σ_v is at most 11LB_v , and (2) the time spent by each object $i \in L_v$ in the vehicle is at most $2 \kappa(s_i, v)$. Similarly for instance \mathcal{I}_{dest} , we obtain a *non-preemptive* tour τ_v that moves all objects in L_v from vertex v to their destinations, such that (1) the length of τ_v is at most 11LB_v , and (2) the time spent by each object $i \in L_v$ in the vehicle is at most $2 \kappa(v, t_i)$. Concatenating these two tours, we obtain that $\sigma_v \cdot \tau_v$ is a *1-preemptive* tour servicing L_v , of length at most $22 \cdot \text{LB}(v)$, where each object $i \in L_v$ spends at most $2(\kappa(s_i, v) + \kappa(v, t_i)) = 2 \cdot \kappa(s_i, t_i)$ time in the vehicle.

We now obtain a depth-first-search traversal (DFS) on \mathcal{T} (restricted to the end-points of demands D_p) to visit all vertices v in level p that have some demand in their subtree (i.e. $L_v \neq \emptyset$), and use the algorithm described above for servicing demands L_v when v is visited in the DFS. This is the tour servicing D_p . Note that the length of the DFS is at most the Steiner lower bound on κ , so it is at most $2 \widetilde{\text{LB}}$. Thus the tour servicing D_p has length at most $2 \widetilde{\text{LB}} + 22 \sum_v \text{LB}(v)$, where v ranges over all vertices in level p of \mathcal{T} . Recall that $\sum_v \text{LB}(v) \leq \widetilde{\text{LB}}$, so the tour servicing D_p has length at most $24 \cdot \widetilde{\text{LB}}$. Additionally, the time spent by each object $i \in D_p$ in the vehicle is at most $2 \kappa(s_i, t_i)$.

Finally concatenating the tours for each level $p = 1, \dots, l$, we obtain a 1-preemptive tour on \mathcal{T} of length $O(\log n) \cdot \text{LB}$. Additionally, the time spent by each object i in the vehicle is at most $2 \cdot \kappa(s_i, t_i)$. This translates to a 1-preemptive tour on the original metric having *expected* length $O(\log^2 n) \cdot \text{LB}$, and with $E[\sum_{i \in D} T_i] \leq \sum_{i \in D} 2 \cdot E[\kappa(s_i, t_i)] \leq O(\log n) \sum_{i \in D} d(s_i, t_i)$, where T_i is the time spent by object i in the vehicle. Using Markov inequality and a union bound, we can ensure that the resulting 1-preemptive tour on metric d satisfies *both* conditions 1-2 with constant probability. ■

3.2 Algorithm for Capacitated mDaR

We are now ready to present our algorithm for capacitated multi-vehicle Dial-a-Ride. The algorithm for mDaR relies on a partial coverage algorithm **Partial** that given subsets $Q \subseteq [q]$ of vehicles and $D \subseteq [m]$ of demands, outputs a schedule for Q of near-optimal makespan that covers some *fraction* of demands in D . The main steps in **Partial** are as follows.

1. Obtain a *single-vehicle* tour satisfying 1-preemptive and bounded-delay properties (Theorem 9).
2. Randomly partition the single vehicle tour into $|Q|$ equally spaced pieces.
3. Solve a matching problem to assign *some* of these pieces to vehicles of Q , so as to satisfy a subset of demands in D .
4. A suitable fraction of unsatisfied demands in D are covered recursively by unused vehicles of Q .

The algorithm first guesses the optimal makespan B of the given instance of mDaR (it suffices to know B within a constant factor, which is required for a polynomial-time algorithm). Let parameter $\alpha := 1 - \frac{1}{1+\lg m}$. For any subset $P \subseteq [q]$, we abuse notation and use P to denote both the set of vehicles P and the multi-set of depots corresponding to vehicles P .

We give an algorithm **Partial** that takes as input a tuple $\langle Q, D, B \rangle$ where $Q \subseteq [q]$ is a subset of vehicles, $D \subseteq [m]$ a subset of demands and $B \in \mathbb{R}_+$, with the *promise* that vehicles Q (originating at their respective depots) suffice to completely serve the demands D at a makespan of B . Given such a promise, **Partial** $\langle Q, D, B \rangle$ returns a schedule of makespan $O(\log n \log m) \cdot B$ that serves a good fraction of D . Algorithm **Partial** $\langle Q, D, B \rangle$ is given in Figure 3. We set parameter $\rho = \Theta(\log n \log m)$, the precise constant in the Θ -notation comes from the analysis.

Lemma 10 *If there exists a schedule of vehicles Q covering all demands D , having makespan at most B , then **Partial** invoked on $\langle Q, D, B \rangle$ returns a schedule of vehicles Q of makespan at most $(16+16\rho) \cdot B$ that covers at least an $\alpha^{\lg z}$ fraction of D , where $z := \min\{|Q|, 2m\} \leq 2m$.*

The final algorithm invokes **Partial** iteratively until all demands are covered: each time with the entire set $[q]$ of vehicles, all uncovered demands, and bound B . If $D \subseteq [m]$ is the set of uncovered demands at any iteration, Lemma 10 implies that **Partial** $\langle [q], D, B \rangle$ returns a schedule of makespan $O(\log m \log n) \cdot B$ that serves at least $\frac{1}{4}|D|$ demands. Hence a standard set-cover analysis implies that all demands will be covered in $O(\log m)$ rounds, resulting in a makespan of $O(\log^2 m \log n) \cdot B$.

Proof of Lemma 10. We proceed by induction on the number $|Q|$ of vehicles. The base case $|Q| = 1$ is easy: the tour τ in Step (2) has length $O(\log^2 n) \cdot B \leq \rho B$, and satisfies all demands (i.e. fraction 1). In the following, we prove the inductive step, when $|Q| \geq 2$.

Preprocessing. Suppose Step (1) applies. Note that $d(V_1, V_2) > B$ and hence there is no demand with source in one of $\{V_1, V_2\}$ and destination in the other. So demands D_1 and D_2 partition D . Furthermore in the optimal schedule, vehicles Q_j (any $j = 1, 2$) only visit vertices in V_j (otherwise the makespan would be greater than B). Thus the two recursive calls to **Partial** satisfy the assumption: there is some schedule of vehicles Q_j serving D_j having makespan B . Inductively, the schedule returned by **Partial** for each $j = 1, 2$ has makespan at most $(16 + 16\rho) \cdot B$ and covers at least $\alpha^{\lg c} \cdot |D_j|$ demands from D_j , where $c \leq \min\{|Q| - 1, 2m\} \leq z$. The schedules returned by the two recursive calls to **Partial** can clearly be run in parallel and this covers at least $\alpha^{\lg z}(|D_1| + |D_2|)$ demands, i.e. an $\alpha^{\lg z}$ fraction of D . So we have the desired performance in this case.

Input: Vehicles $Q \subseteq [q]$, demands $D \subseteq [m]$, bound $B \geq 0$ such that Q can serve all demands in D at makespan B .

Preprocessing

1. If the minimum spanning tree (MST) on vertices Q contains an edge of length greater than $3B$, there is a non-trivial partition $\{Q_1, Q_2\}$ of Q with $d(Q_1, Q_2) > 3B$. For $j \in \{1, 2\}$, let $V_j = \{v \in V \mid d(Q_j, v) \leq B\}$ and D_j be all demands of D induced on V_j . Run in parallel the schedules from $\text{Partial}\langle Q_1, D_1, B \rangle$ and $\text{Partial}\langle Q_2, D_2, B \rangle$. Assume this is not the case in the following.

Random partitioning

2. Obtain single-vehicle 1-preemptive tour τ using capacity k and serving demands D , by applying Theorem 9.
3. Choose a uniformly random offset $\eta \in [0, \rho B]$ and cut edges of tour τ at distances $\{p\rho B + \eta \mid p = 1, 2, \dots\}$ along the tour to obtain a set \mathcal{P} of pieces of τ .
4. C'' is the set of objects $i \in D$ such that i is carried by the vehicle in τ over some edge that is cut in Step (3); and $C' := D \setminus C''$. Ignore C'' objects in the rest of the algorithm.

Load rebalancing

5. Construct bipartite graph H with vertex sets \mathcal{P} and Q and an edge between piece $P \in \mathcal{P}$ and depot $f \in Q$ iff $d(f, P) \leq 2B$. For any subset $A \subseteq \mathcal{P}$, $\Gamma(A) \subseteq Q$ denotes the neighborhood of A in graph H . Let $\mathcal{S} \subseteq \mathcal{P}$ be any *maximal* set that satisfies $|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{S}|}{2}$.
6. Compute a 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \rightarrow Q \setminus \Gamma(\mathcal{S})$, i.e. function s.t. $(P, \pi(P))$ is an edge in H for all $P \in \mathcal{P} \setminus \mathcal{S}$, and the number of pieces mapping to any $f \in Q \setminus \Gamma(\mathcal{S})$ is $|\pi^{-1}(f)| \leq 2$.

Recursion

7. Define $C_1 := \{i \in C' \mid \text{either } s_i \in \mathcal{S} \text{ or } t_i \in \mathcal{S}\}$; and $C_2 := C' \setminus C_1$.
8. Run in parallel the *recursive* schedule $\text{Partial}\langle \Gamma(\mathcal{S}), C_1, B \rangle$ for C_1 and the following for C_2 :
 - (a) Each vehicle $f \in Q \setminus \Gamma(\mathcal{S})$ traverses the pieces $\pi^{-1}(f)$, moving all C_2 -objects in them from their source to preemption-vertex, and returns to its depot.
 - (b) Each vehicle $f \in Q \setminus \Gamma(\mathcal{S})$ again traverses the pieces $\pi^{-1}(f)$, this time moving all C_2 -objects in them from their preemption-vertex to destination, and returns to its depot.

Output: A schedule of Q of makespan $(16 + 16\rho) \cdot B$ that serves an $\alpha^{\lg \min\{|Q|, 2m\}}$ fraction of D .

Figure 3: Algorithm $\text{Partial}\langle Q, D, B \rangle$ for capacitated mDaR.

Random partitioning. The harder part of the algorithm is when Step (1) does not apply: so the MST length on Q is at most $3|Q| \cdot B$. Note that when the depots Q are contracted to a single vertex, the MST on the end-points of D plus the contracted depot-vertex has length at most $|Q| \cdot B$ (the optimal makespan schedule induces such a tree). Thus the MST on the depots Q along with end-points of D has length at most $4|Q| \cdot B$. Based on the assumption in Lemma 10 and the flow lower bound for mDaR, we have $\sum_{i \in D} d(s_i, t_i) \leq k|Q| \cdot B$. It follows that for the *single vehicle* Dial-a-Ride instance solved in Step (2), the Steiner and flow lower-bounds (denoted LB in Theorem 9) are $O(1) \cdot |Q|B$. Theorem 9 now implies

that (with high probability) τ is a 1-preemptive tour servicing D , of length at most $O(\log^2 n)|Q| \cdot B$ such that $\sum_{i \in D} T_i \leq O(\log n) \cdot |D|B$, where T_i denotes the total time spent in the vehicle by demand $i \in D$. The bound on the delay uses the fact that $\max_{i=1}^m d(s_i, t_i) \leq B$.

Choosing a large enough constant corresponding to $\rho = \Theta(\log n \log m)$, the length of τ is upper bounded by $\rho|Q| \cdot B$ (since $n \leq 2m$). So the cutting procedure in Step (3) results in at most $|Q|$ pieces of τ , each of length at most $2\rho B$. The objects $i \in C''$ (as defined in Step (4)) are called a *cut objects*. We restrict attention to the other objects $C' = D \setminus C''$ that are not ‘cut’. For each object $i \in C'$, the path traced by it (under single vehicle tour τ) from its source s_i to preemption-point and the path from its preemption-point to t_i are both completely contained in pieces of \mathcal{P} . Figure 4 gives an example of objects in C' and C'' , and the cutting procedure.

Claim 11 *The expected number of objects in C'' is at most $\sum_{i \in D} \frac{T_i}{\rho B} \leq O(\frac{1}{\log m}) \cdot |D|$.*

Proof: The probability (over choice of η) that object $i \in D$ is cut equals $\frac{T_i}{\rho B}$ where T_i is the total time spent by i in tour τ . The claim follows by linearity of expectation and $\sum_{i \in D} T_i \leq O(\log n) \cdot |D|B$. ■

We can derandomize Step (3) and pick the best offset η (there are at most polynomially many combinatorially distinct offsets). Claim 11 implies (again choosing large enough constant in $\rho = \Theta(\log n \log m)$) that $|C'| \geq (1 - \frac{1}{2 \log m})|D| \geq \alpha \cdot |D|$ demands are *not cut*; note that this occurs with probability one since we use the best choice for η . Now onwards we only consider the set C' of uncut demands. Let \mathcal{P} denote the pieces obtained by cutting τ as above, recall $|\mathcal{P}| \leq |Q|$. A piece $P \in \mathcal{P}$ is said to be non-trivial if the vehicle in the 1-preemptive tour τ carries some C' -object while traversing P . Note that the number of non-trivial pieces in \mathcal{P} is at most $2|C'| \leq 2m$: each C' -object appears in at most 2 pieces, one where it is moved from source to preemption-vertex and other from preemption-vertex to destination. Retain only the non-trivial pieces in \mathcal{P} ; so $|\mathcal{P}| \leq \min\{|Q|, 2m\} = z$. The pieces in \mathcal{P} may not be one-to-one assignable to the depots since the algorithm has not taken the depot locations into account. We determine which pieces may be assigned to depots by considering a matching problem between \mathcal{P} and the depots in Step (5) and (6).

Load rebalancing. The bipartite graph H (defined in Step (5)) represents which pieces and depots may be assigned to each other. Piece $P \in \mathcal{P}$ and depot $f \in Q$ are assignable iff $d(f, P) \leq 2B$, and in this case graph H contains an edge (P, f) . We claim that corresponding to the ‘maximal contracting’ set \mathcal{S} (defined in Step (5)), the 2-matching π (in Step (6)) is guaranteed to exist. Note that $|\Gamma(\mathcal{S})| \leq \frac{|\mathcal{S}|}{2}$, but $|\Gamma(\mathcal{T})| > \frac{|\mathcal{T}|}{2}$ for all $\mathcal{T} \supset \mathcal{S}$. For any $T' \subseteq \mathcal{P} \setminus \mathcal{S}$, let $\tilde{\Gamma}(T')$ denote the neighborhood of T' in $Q \setminus \Gamma(\mathcal{S})$. The maximality of \mathcal{S} implies: for any non-empty $T' \subseteq \mathcal{P} \setminus \mathcal{S}$, $\frac{|\mathcal{S}|}{2} + \frac{|T'|}{2} = \frac{|\mathcal{S} \cup T'|}{2} < |\Gamma(\mathcal{S} \cup T')| = |\Gamma(\mathcal{S})| + |\tilde{\Gamma}(T')|$, i.e. $|\tilde{\Gamma}(T')| \geq \frac{|T'|}{2}$. Hence by *Hall’s condition*, there is a 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \rightarrow Q \setminus \Gamma(\mathcal{S})$. The set \mathcal{S} and 2-matching π can be easily computed in polynomial time.

Recursion. In Step (7), demands C' are further partitioned into two sets: C_1 consists of objects that are *either* picked-up *or* dropped-off in some piece of \mathcal{S} ; and C_2 -objects are picked-up *and* dropped-off in pieces of $\mathcal{P} \setminus \mathcal{S}$. The vehicles $\Gamma(\mathcal{S})$ suffice to serve all C_1 objects, as shown below.

Claim 12 *There exists a schedule of vehicles $\Gamma(\mathcal{S})$ serving demands C_1 , having makespan B .*

Proof: Consider the schedule of makespan B that serves all demands $C' = C_1 \cup C_2$ using vehicles Q : this is implied by the promise on instance $\langle Q, D, B \rangle$. We claim that in this schedule, no vehicle from $Q \setminus \Gamma(\mathcal{S})$ moves any C_1 object. Suppose (for a contradiction) that the vehicle from depot $f \in Q \setminus \Gamma(\mathcal{S})$ moves object $i \in C_1$ at some point in this schedule; then it must be that $d(f, s_i)$ and $d(f, t_i) \leq 2B$. But since $i \in C_1$, at least one of s_i or t_i is in a piece of \mathcal{S} , and this implies that there is some piece $P \in \mathcal{S}$ with $d(f, P) \leq 2B$, i.e. $f \in \Gamma(\mathcal{S})$, which is a contradiction! Thus the only vehicles participating in the movement of C_1 objects are $\Gamma(\mathcal{S})$, which implies the claim. ■

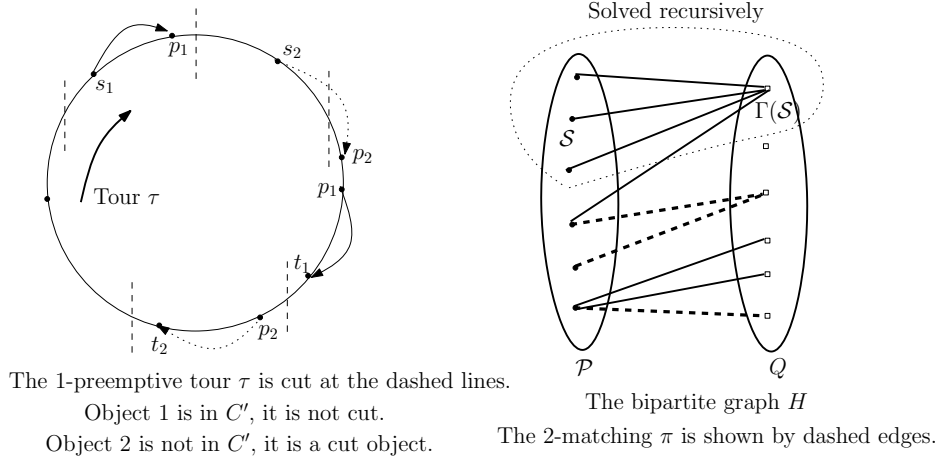


Figure 4: Cutting and patching steps in algorithm **Partial**.

In the final schedule, a *large fraction* of C_1 demands are served by vehicles $\Gamma(S)$, and *all* the C_2 demands are served by vehicles $Q \setminus \Gamma(S)$. Figure 4 shows an example of this partition.

Serving C_1 demands. Based on Claim 12, the recursive call **Partial** $\langle \Gamma(S), C_1, B \rangle$ (made in Step (8)) satisfies the assumption required in Lemma 10. Since $|\Gamma(S)| \leq \frac{|\mathcal{P}|}{2} \leq \frac{|Q|}{2} < |Q|$, we obtain inductively that **Partial** $\langle \Gamma(S), C_1, B \rangle$ returns a schedule of makespan $(16 + 16\rho) \cdot B$ covering at least $\alpha^{\lg y} \cdot |C_1|$ demands of C_1 , where $y = \min\{|\Gamma(S)|, 2m\}$. Note that $y \leq |\Gamma(S)| \leq |\mathcal{P}|/2 \leq z/2$ (recall $|\mathcal{P}| \leq z$), which implies that at least $\alpha^{\lg z-1} |C_1|$ demands are covered.

Serving C_2 demands. These are served by vehicles $Q \setminus \Gamma(S)$ using the 2-matching π , in two rounds as specified in Step (8). This suffices to serve all objects in C_2 since for any $i \in C_2$, the paths traversed by object i under τ , namely $s_i \rightsquigarrow p_i$ (its preemption-point) and $p_i \rightsquigarrow t_i$ are contained in pieces of $\mathcal{P} \setminus S$. Furthermore, since $|\pi^{-1}(f)| \leq 2$ for all $f \in Q \setminus \Gamma(S)$, the distance traveled by vehicle f in one round is at most $2 \cdot 2(2B + 2\rho B)$. So the time taken by this schedule is at most $2 \cdot 4(2B + 2\rho B) = (16 + 16\rho) \cdot B$.

The schedule of vehicles $\Gamma(S)$ (serving C_1) and vehicles $Q \setminus \Gamma(S)$ (serving C_2) can clearly be run in parallel. This takes time $(16 + 16\rho) \cdot B$ and covers in total at least $|C_2| + \alpha^{\lg z-1} |C_1| \geq \alpha^{\lg z-1} |C'| \geq \alpha^{\lg z} |D|$ demands of D . This completes the proof of the inductive step of Lemma 10.

Using Lemma 10 repeatedly as mentioned earlier, we obtain an $O(\log^2 m \cdot \log n)$ -approximation algorithm. Using some preprocessing steps (described in the next subsection), we have Theorem 3.

3.3 Weighted mDaR

The multi-vehicle Dial-a-Ride problem as defined assumes that all objects have the same ‘weight’, i.e. each object occupies a unit capacity. In the weighted mDaR problem, each object $i \in [m]$ also has a weight w_i , and the capacity constraint requires that no vehicle carry a total weight of more than k at any time. In this section, we obtain an $O(\log^3 n)$ -approximation algorithm for weighted mDaR. In particular this would imply Theorem 3.

The algorithm for weighted mDaR first guesses the optimal makespan B of the given instance. It serves the demands in two phases: the 1st phase involves pre-processing ‘heavy demands’ and has a makespan of $O(1) \cdot B$; the 2nd phase is identical to the algorithm **Partial** in Section 3.2 and covers all remaining demands. For every $u, v \in V$ let $\text{dem}_{u,v}$ denote the total weight of objects having source u and destination v . Define $H = \{(u, v) \in V \times V \mid \text{dem}_{u,v} \geq k/2\}$ to be the *heavy* vertex-pairs, and \hat{H} the set of demands between pairs of H .

Phase I. In this pre-processing step, we cover all demands \hat{H} between heavy vertex-pairs. The algorithm **PreProc** for solving instance \mathcal{U} and its analysis follow along the lines of algorithm **Partial** in Section 3.2; in fact it is much easier. Below we use the same notation $P \subseteq [q]$ for a set P of vehicles and the multi-set of depots corresponding to P . The algorithm **PreProc** will satisfy the following property.

Lemma 13 *Given any subset $Q \subseteq [q]$ of vehicles and $D \subseteq H$ of heavy vertex-pairs and bound B such that there is a schedule of vehicles Q covering demands \hat{D} with makespan at most B , **PreProc** $\langle Q, D, B \rangle$ returns a schedule of vehicles Q covering demands \hat{D} that has makespan $O(1) \cdot B$.*

Note: Invoking **PreProc** $\langle [q], H, B \rangle$ gives the desired schedule covering \hat{H} at makespan $O(1) \cdot B$.

Algorithm **PreProc** first ensures that each edge of the minimum spanning tree on Q has length at most $3B$, otherwise the problem decouples into two disjoint smaller problems (as in Step (1) of **Partial**). Hence, as in algorithm **Partial**, we also obtain that:

$$\text{The MST on all sources/destinations in } D \text{ has length at most } O(1) \cdot |Q|B \quad (1)$$

Demands with source u and destination v are referred to as (u, v) demands. For every $(u, v) \in D$, using a greedy procedure, one can partition all (u, v) demands such that the total weight of each part (except possibly the last) is between $\frac{k}{2}$ and k . For any $(u, v) \in D$, let $g_{u,v}$ denote the number of parts in this partition of (u, v) demands; note that $g_{u,v} \leq \frac{2}{k} \text{dem}_{u,v} + 1 \leq \frac{4}{k} \text{dem}_{u,v}$ (since $\text{dem}_{u,v} \geq k/2$). The flow lower bound for the mDaR instance with vehicles Q and demands \hat{D} equals $\frac{1}{|Q|k} \sum_{(u,v) \in D} \text{dem}_{u,v} \cdot d(u, v) \geq \frac{1}{4|Q|} \sum_{(u,v) \in D} g_{u,v} \cdot d(u, v)$. Using the assumption in Lemma 13, we have that:

$$\sum_{(u,v) \in D} g_{u,v} \cdot d(u, v) \leq 4|Q|B. \quad (2)$$

In the rest of algorithm **PreProc**, we will consider each part in the above partition of (u, v) demands (for each $(u, v) \in D$) as a single object of weight k ; so all the objects in one part will be moved together. Scaling down the weights and capacity by k , we obtain the following equivalent *unit-weight unit-capacity instance* \mathcal{U} : for each $(u, v) \in D$ there are $g_{u,v}$ demands with weight 1, source u and destination v , and each vehicle in Q has capacity 1.

Since \mathcal{U} has unit capacity and weights, we can use the *Stacker-crane* algorithm [16] to obtain a single vehicle *non-preemptive* tour τ serving all demands, having length 1.8 times the Steiner and flow lower bounds. For this single vehicle problem, the lower bounds corresponding to \mathcal{U} are: (Steiner bound) TSP on all sources/destinations in D , and (flow bound) $\sum_{(u,v) \in H} g_{u,v} \cdot d(u, v)$. From (1) and (2) above, these are both $O(1) \cdot |Q|B$. The algorithm next cuts tour τ to obtain at most $|Q|$ pieces \mathcal{P} such that each piece has length $O(1) \cdot B$. In addition it can be ensured that *no* object is being carried over the cut edges: since the vehicle carries at most one object at any time and each source-destination distance is at most B . When $|Q| = 1$, the algorithm ends here, and returns this single tour of length $O(1)B$ that serves all demands \hat{D} . This proves the base case of Lemma 13.

When $|Q| \geq 2$, construct bipartite graph \mathcal{H} with vertex sets \mathcal{P} and Q and an edge between piece $P \in \mathcal{P}$ and depot $f \in Q$ iff $d(f, P) \leq 2B$; $\Gamma(A)$ denotes the neighborhood of any $A \subseteq \mathcal{P}$ in \mathcal{H} . The algorithm finds any maximal set $\mathcal{S} \subseteq \mathcal{P}$ with $|\Gamma(\mathcal{S})| < |\mathcal{S}|/2$ (as in Step (5) of **Partial**). This implies a 2-matching $\pi : \mathcal{P} \setminus \mathcal{S} \rightarrow Q \setminus \Gamma(\mathcal{S})$ such that there exists a schedule of vehicles $\Gamma(\mathcal{S})$ serving demands in the pieces \mathcal{S} with makespan B (c.f. Claim 12). Let $C \subseteq D$ denote the heavy demand-pairs served in some piece of \mathcal{S} . The final schedule returned by **PreProc** $\langle Q, D \rangle$ involves: (i) schedule for vehicles $Q \setminus \Gamma(\mathcal{S})$ given by π (covering demand-pairs $D \setminus C$); and (ii) schedule for vehicles $\Gamma(\mathcal{S})$ obtained recursively **PreProc** $\langle \Gamma(\mathcal{S}), C \rangle$ (covering demands C). The recursively obtained schedule has makespan $O(1) \cdot B$ by the induction hypothesis (since $|\Gamma(\mathcal{S})| < |Q|$, and $\langle \Gamma(\mathcal{S}), C \rangle$ satisfies the assumption in Lemma 13); hence the final schedule also has makespan $O(1) \cdot B$, which proves the inductive step.

Phase II. Let $L = \{(u, v) \in V \times V \mid 0 < \text{dem}_{u,v} < k/2\}$ be the *light* vertex-pairs. The algorithm for this phase treats all (u, v) demands as a *single object* of weight $\text{dem}_{u,v}$ from u to v ; so there are $m = |L| \leq n^2$ distinct objects. The algorithm for this case is identical to **Partial** of Section 3.2 for the unweighted case: Theorem 9 generalizes easily to the weighted case, and the Steiner and flow lower-bounds stay the same after combining demands in L . Thus we obtain an $O(\log^2 m \log n) = O(\log^3 n)$ approximate schedule that covers all remaining demands (setting $m \leq n^2$).

Theorem 14 *There is a randomized $O(\log^3 n)$ -approximation algorithm for weighted preemptive mDaR.*

3.4 Improved Guarantee for Metrics Excluding a Fixed Minor

In this section, we show that our algorithm for preemptive mDaR achieves an $O(\log^2 n)$ approximation guarantee when the metric is induced by a graph excluding any fixed minor. The main ingredient is the following improvement in the single vehicle structure from Theorem 9.

Theorem 15 *There is a randomized poly-time algorithm that for any instance of single vehicle dial-a-ride on metrics excluding a fixed minor, computes a 1-preemptive tour τ servicing all demands D that satisfies the following conditions (where LB is the average of the Steiner and flow lower bounds):*

1. **Total length:** $d(\tau) \leq O(\log n) \cdot \text{LB}$.
2. **Bounded delay:** $\sum_{i \in D} T_i \leq O(1) \sum_{i \in D} d(s_i, t_i)$ where T_i is the total time spent by object $i \in D$ in the vehicle under the schedule given by τ .

Using this 1-preemptive tour within the algorithm of Section 3.2 (setting parameter $\rho = \Theta(\log m)$), we immediately obtain Theorem 4. In proving Theorem 15, we first show that metrics induced by (fixed) minor-free graphs allow a so-called γ -separated cover (Subsection 3.4.1). Then we show (in Subsection 3.4.2) how this property can be used to obtain the single-vehicle tour claimed in Theorem 15.

3.4.1 γ -Separated Covers

We are given a metric (V, d) that is induced by a graph $G = (V, E)$ and edge-lengths $w : E \rightarrow \mathbb{Z}_+$. For any pair $u, v \in V$ of vertices, the distance $d(u, v)$ equals the shortest path between u and v under edge-lengths w . We assume that the graph G does not contain any K_r -minor; here r is a fixed parameter. Recall that a *cluster* refers to any subset of vertices. We prove the following.

Theorem 16 *Given a K_r -minor free graph $G = (V, E, w)$ and integer $\gamma \geq 0$, there is a polynomial-time algorithm to compute a collection \mathcal{C} of clusters along with a partition Z_1, \dots, Z_p of \mathcal{C} such that:*

1. $p = 2^{O(r)}$.
2. For each $l \in [p]$ and distinct clusters $A, B \in Z_l$, the distance between A and B , $d(A, B) \geq \gamma$.
3. The diameter of any cluster in $S \in \mathcal{C}$, $\max_{u, v \in S} d(u, v) \leq O(r^2) \cdot \gamma$.
4. For any pair $u, v \in V$ of vertices with $d(u, v) \leq \gamma$, there is some cluster $A \in \mathcal{C}$ having $u, v \in A$.

Such a collection \mathcal{C} is called γ -separated cover.

The proof of this theorem essentially follows from the KPR decomposition algorithm [22]. In fact without conditions (1-2), it is implied by Theorem 7. Achieving conditions (1-2) requires a further modification to the KPR decomposition, as described below.

We first remind the reader of the main property of the KPR decomposition [22]. We will consider graph G as having unit length edges, by subdividing each edge $(u, v) \in E$ to become a path of length w_{uv} between u and v (this also increases the number of vertices). The distance function d remains unchanged by this modification.

Definition 17 (Theorem 4.2, [22]) Let $G_1 = G$ and $\delta \in \mathbb{Z}_+$ a distance parameter. Let G_1, G_2, \dots, G_{r+1} be any sequence of subgraphs of G , where each G_{i+1} is obtained from G_i as follows:

1. Construct a breadth-first-search tree in G_i from an arbitrary root-vertex.
2. Select any set of δ consecutive levels in this breadth-first-search tree, and let G_{i+1} be any connected component of the subgraph in G induced by these levels.

Then, the diameter of G_{r+1} is $O(r^2) \cdot \delta$.

We are now ready to proceed with the proof of Theorem 16. We give a recursive procedure to generate the desired γ -separated cover \mathcal{C} of G . We initialize $\mathcal{C} = \emptyset$, and invoke $\text{Split}\langle G, V, 0, \phi \rangle$. Algorithm Split takes as input: an induced subgraph $H = (V(H), E(H))$ of G , terminals $T \subseteq V(H)$, depth of recursion i , and color $\tau \in \{0, 1, 2\}^i$. Then it does the following.

1. If $i = r$ then add tuple $\langle T, \tau \rangle$ to collection \mathcal{C} , and stop.
2. Construct a BFS tree from any vertex of H , and enumerate the levels starting from zero (at the root).
3. For each $j \in \{0, 1, 2\}$, integer $l \geq 0$, and connected component H' in the subgraph of G induced by levels $\{(3l + j - 1)\gamma, \dots, (3l + j + 3)\gamma\}$ of BFS do:
 - (a) Set $T' \leftarrow$ vertices of $T \cap V(H')$ in levels $\{(3l + j)\gamma, \dots, (3l + j + 2)\gamma\}$ of BFS.
 - (b) Set $\tau' \leftarrow \tau$ concatenated with j .
 - (c) Recurse on $\text{Split}\langle H', T', i + 1, \tau' \rangle$.

At the end of the algorithm, \mathcal{C} consists of several tuples of the form $\langle C, \tau \rangle$ where C is a cluster and $\tau \in \{0, 1, 2\}^r$ is its color. This naturally corresponds to partitioning \mathcal{C} into $p = 3^r$ parts: clusters of the same color form a part. We now show that \mathcal{C} satisfies the conditions in Theorem 16. Condition 1 clearly holds by the construction.

Condition 4. Let $u, v \in V$ be such that $d(u, v) \leq \gamma$, and let $P \subseteq V$ denote the vertices on any shortest $u - v$ path in G (note that $|P| \leq \gamma + 1$). We claim inductively that for each $i \in \{0, \dots, r\}$, there is some call to Split at depth i such that all vertices of P appear as terminals. The base case $i = 0$ is obvious since all vertices are terminals. Assuming the claim to be true for depth i , we prove it for $i + 1$. Let $\langle H, T, i, \tau \rangle$ denote the call to Split at depth i where $P \subseteq T \subseteq V(H)$. Since the P -vertices define a path of length at most γ (in G and also H), all P -vertices appear in some γ consecutive levels of any BFS on H . Thus there exist values $j \in \{0, 1, 2\}$ and $l \in \mathbb{Z}_+$ such that P is contained in levels $\{(3l + j)\gamma, \dots, (3l + j + 2)\gamma\}$ of the BFS on H . This implies that one of the recursive calls (corresponding to this value of j and l) contains P as terminals. Using this claim when $i = r$ implies that there is some cluster in \mathcal{C} containing both u and v (in fact containing the entire path P).

Condition 3. This is a direct consequence of the KPR decomposition. Note that every subgraph at depth r of algorithm Split is obtained from G via the sequence of operations in Definition 17 with $\delta = 4\gamma + 1$. Thus each such subgraph has diameter $O(r^2) \cdot \gamma$. Finally, any cluster in \mathcal{C} is a subset of some subgraph at depth r of Split ; so it also has diameter $O(r^2) \cdot \gamma$.

Condition 2. This well-separated property is the main reason for the use of ‘terminals’ in algorithm Split and the decomposition defined in Step (3). Let A, B be any two distinct clusters in \mathcal{C} having the same color: we will show that $d(A, B) \geq \gamma$. Each of A and B corresponds to a “root-leaf path” in the recursion tree for algorithm Split (where the root is $\text{Split}\langle G, V, 0, \phi \rangle$). Consider the call $\langle H, T, i, \tau \rangle$ to Split after which the paths for A and B diverge; let $\alpha_a = \langle H_a, T_a, i + 1, \tau' \rangle$ and $\alpha_b = \langle H_b, T_b, i + 1, \tau' \rangle$ denote the respective recursive calls from $\langle H, T, i, \tau \rangle$ where $A \subseteq T_a$ and $B \subseteq T_b$ (note that both have the same color τ' since A and B have the same color at the end of the algorithm). We will show that $d(T_a, T_b) \geq \gamma$ which in particular implies $d(A, B) \geq \gamma$. Note that both α_a and α_b are generated by the same value $j \in \{0, 1, 2\}$, since they have the same color. Let α_a (resp. α_b) correspond to value $l = l_a$ (resp. $l = l_b$). Consider the following cases:

1. $l_a \neq l_b$. In this case, we show that the terminal-sets T_a and T_b are far apart in the BFS on H . Observe that T_a appears within levels $\{(3l_a + j)\gamma, \dots, (3l_a + j + 2)\gamma\}$; and T_b within $\{(3l_b + j)\gamma, \dots, (3l_b + j + 2)\gamma\}$. If $l_a < l_b$ (the other case is identical), then $3l_b + j \geq 3l_a + j + 3$; i.e. T_a and T_b are separated by at least $\gamma - 1$ levels in the BFS of H . Using Claim 18 below, we have $d(T_a, T_b) \geq \gamma$ in G ; otherwise H contains a path from T_a to T_b of length at most $\gamma - 1$, meaning that T_a and T_b are separated by $\leq \gamma - 2$ levels in the BFS, a contradiction!
2. $l_a = l_b = l$. In this case, it must be that H_a and H_b are two *disconnected components* of the subgraph (of G) induced on levels $\{(3l + j - 1)\gamma, \dots, (3l + j + 3)\gamma\}$ of the BFS on H . Furthermore, T_a and T_b both appear within levels $\{(3l + j)\gamma, \dots, (3l + j + 2)\gamma\}$. Again by Claim 18, we must have $d(T_a, T_b) \geq \gamma$ in G ; otherwise H contains a path from T_a to T_b of length at most $\gamma - 1$, which would mean that H_a and H_b are connected within levels $\{(3l + j - 1)\gamma, \dots, (3l + j + 3)\gamma\}$.

Claim 18 For every call $\text{Split}\langle H, T, i, \tau \rangle$, and terminals $x, y \in T$, if $d(x, y) \leq \gamma$ (in graph G) then H contains an $x - y$ path of length at most $d(x, y)$.

Proof: We proceed by induction on i . The claim is obviously true for the call $\text{Split}\langle G, V, 0, \phi \rangle$ at depth 0. Assuming the claim for any depth i call $\text{Split}\langle H, T, i, \tau \rangle$, we prove it for any call $\text{Split}\langle H', T', i + 1, \tau' \rangle$ generated by it. Let $x, y \in T'$ with $d(x, y) \leq \gamma$. Clearly $x, y \in T$, and by the induction hypothesis, H contains an $x - y$ path π of length at most $d(x, y) \leq \gamma$. It suffices to show that π is also contained in H' . Let $j \in \{0, 1, 2\}$ and $l \in \mathbb{Z}_+$ denote the values that generated the recursive call $\text{Split}\langle H', T', i + 1, \tau' \rangle$. Since $x, y \in T'$, both these vertices lie within levels $\{(3l + j)\gamma, \dots, (3l + j + 2)\gamma\}$ of the BFS on H . Furthermore, π is an $x - y$ path in H of length at most γ ; so π is contained within levels $\{(3l + j - 1)\gamma, \dots, (3l + j + 3)\gamma\}$ and so it lies in the connected component H' that contains x and y . ■

Running time. It is easy to see that each call to Split generates only polynomial number of recursive calls. Since the depth of the recursion is at most r , the running time is polynomial for fixed r .

This completes the proof of Theorem 16.

We note that the collection \mathcal{C} also satisfies the ‘sparse cover’ property (i.e. condition (3) in Theorem 7), namely each vertex $v \in V$ appears in at most $O(2^r)$ clusters. It is easy to show (inductively) that the number of depth i calls to Split where any vertex v appears as a terminal is at most 2^i . However this property is not required in the following proof of Theorem 15.

3.4.2 Algorithm for Theorem 15

Recall that we are given an instance of single-vehicle Dial-a-Ride on a metric (V, d) induced by an edge-weighted K_r -minor free graph $G = (V, E, w)$; here $w : E \rightarrow \mathbb{Z}_+$ denotes the edge-lengths and d the shortest-path distances under w . The vehicle has capacity k and $\{s_i, t_i\}_{i \in D}$ are the demand-pairs. Again r is a fixed constant. LB denotes the average of the Steiner (i.e. minimum TSP tour on all sources and destinations) and flow lower-bounds (i.e. $\frac{1}{k} \sum_{i \in D} d(s_i, t_i)$) for this instance.

Let Δ denote the diameter of the metric; by standard scaling arguments, we may assume WLOG that Δ is at most polynomial in $n = |V|$. We group the demands D into $\lceil \log_2 \Delta \rceil$ groups based on the source-destination distances: For each $j = 1, \dots, \lceil \log_2 \Delta \rceil$, group G_j consists of those demands $i \in D$ with $2^{j-1} \leq d(s_i, t_i) \leq 2^j$. We will show how to service each group G_j separately at a cost of $O(1) \cdot \text{LB}$ such that the time spent by each object $i \in G_j$ in the vehicle is $T_i = O(2^j) = O(1) \cdot d(s_i, t_i)$. Since the number of groups is $O(\log n)$, this would imply the theorem.

Serving group G_j Set distance parameter $\gamma = 2^j$, and obtain a γ -separated cover \mathcal{C} along with its partition $\{Z_1, \dots, Z_p\}$, using Theorem 16. Each demand $i \in G_j$ is *assigned* to some cluster of \mathcal{C} containing both its end-points: since $d(s_i, t_i) \leq \gamma$, this is well-defined by property (4) of Theorem 15. We further partition demands in G_j into H_1, \dots, H_p where each H_l contains all demands assigned to

clusters of Z_l . The algorithm will serve demands in each H_l separately by a tour of length $O(1) \cdot \text{LB}$ that also satisfies the bounded delay condition 2. This suffices to prove the theorem since $p = O(1)$ by property (1) of Theorem 16.

Serving H_l . For any cluster $S \in Z_l$ let $B(S) \subseteq G_j$ denote the demands assigned to S . Let $\text{LB}(S)$ denote the average of the Steiner and flow lower bounds restricted to demands $B(S)$. Any cluster $S \in Z_l$ is called non-trivial if $B(S) \neq \emptyset$. Also pick an arbitrary vertex in each cluster as its *center*. Let \mathcal{T} denote a 1.5-approximate TSP tour containing the centers of all non-trivial clusters of Z_l ; this can be computed in polynomial time [11].

Claim 19 *We have $\sum_{S \in Z_l} \text{LB}(S) \leq O(1) \cdot \text{LB}$, and $d(\mathcal{T}) \leq O(1) \cdot \text{LB}$.*

Proof: It is clear that the sum of the flow lower-bounds over all clusters $S \in Z_l$ is at most the flow lower-bound on demands $H_l \subseteq D$. Let $\text{Tsp}(S)$ denote the minimum length TSP on the end points of demands $B(S)$, i.e. the Steiner lower bound on S . We show below that $\sum_{S \in Z_l} \text{Tsp}(S) \leq O(1) \cdot \text{Tsp}$ where Tsp denotes the minimum TSP on end points of $H_l \subseteq D$ (i.e. Steiner lower bound on H_l). This would prove the first part of the claim.

For any cluster $S \in Z_l$, let $n(S)$ denote the number of connected segments in $\text{Tsp} \cap S$ (i.e. Tsp restricted to S), and $C_1^S, \dots, C_{n(S)}^S$ denote these segments. Note that tour Tsp travels from one cluster in Z_l to another at least $\sum_{S \in Z_l} n(S)$ times. By property (2) of Theorem 15, the distance between any two distinct clusters in Z_l is at least γ . So we obtain:

$$\text{Tsp} \geq \gamma \sum_{S \in Z_l} n(S) \quad (3)$$

We define a TSP tour on end-points of demands $B(S)$ by connecting the end points of segments $C_1^S, \dots, C_{n(S)}^S$. This requires $n(S)$ new edges, each of length at most $O(\gamma)$ (i.e. diameter of cluster S , by property (3) of Theorem 15). So the minimum TSP tour for any $S \in Z_l$ has length $\text{Tsp}(S) \leq \sum_{a=1}^{n(S)} d(C_a^S) + O(\gamma) \cdot n(S)$. Hence,

$$\sum_{S \in Z_l} \text{Tsp}(S) \leq \sum_{S \in Z_l} \sum_{a=1}^{n(S)} d(C_a^S) + O(\gamma) \sum_{S \in Z_l} n(S) \leq \text{Tsp} + O(\gamma) \sum_{S \in Z_l} n(S) \leq O(1) \cdot \text{Tsp}.$$

Above, the second inequality follows from the fact that the C_l^S s are disjoint segments in Tsp , and the last inequality uses (3).

To obtain the second part of the claim, augment Tsp as follows. For each non-trivial cluster $S \in Z_l$, add two edges to the center of S from some vertex in $\text{Tsp} \cap S$. This gives a TSP tour visiting the centers of all non-trivial clusters in Z_l . Note that the number of non-trivial clusters of Z_l is at most $\sum_{S \in Z_l} n(S)$ since $n(S) \geq 1$ for all non-trivial $S \in Z_l$. Since the diameter of each $S \in Z_l$ is $O(\gamma)$ (property (3) of Theorem 15), the increase in length of Tsp is at most $2 \cdot O(\gamma) \sum_{S \in Z_l} n(S) \leq O(1) \cdot \text{Tsp}$ by (3). So there is a TSP tour on the centers of Z_l 's non-trivial clusters having length at most $O(1) \cdot \text{LB}$. ■

For each $S \in Z_l$, the demands $B(S)$ assigned to it are served separately. The flow lower-bound on S is at least $\frac{\gamma}{2k} |B(S)|$ (recall that each demand $i \in G_j$ has $d(s_i, t_i) \geq \gamma/2$). Consider an instance \mathcal{I}_{src} of CVRP with all sources from $B(S)$ and S 's center as the common destination. Since the diameter of S is $O(\gamma)$, the flow lower bound of \mathcal{I}_{src} is at most $O(\gamma) \cdot \frac{|B(S)|}{k} \leq O(1) \cdot \text{LB}(S)$. The Steiner lower bound of \mathcal{I}_{src} is also $O(1) \cdot \text{LB}(S)$. So Theorem 8 (with delay parameter $\beta = 2$) implies a non-preemptive tour $\tau_{src}(S)$ that moves all objects from sources of $B(S)$ to the center of S , having length $O(1) \cdot \text{LB}(S)$ such that each object spends at most $O(1) \cdot \gamma$ time in the vehicle. Similarly, we can obtain non-preemptive tour $\tau_{dest}(S)$ that moves all objects from the center of S to destinations of $B(S)$, having length $O(1) \cdot \text{LB}(S)$ such that each object spends at most $O(1) \cdot \gamma$ time in the vehicle. Concatenating

$\tau_{src}(S)$ and $\tau_{dest}(S)$ gives a 1-preemptive tour serving demands in S having length $O(1) \cdot \text{LB}(S)$ such that each object $i \in B(S)$ spends at most $O(\gamma) \leq O(1) \cdot d(s_i, t_i)$ time in the vehicle.

The final solution for demands H_l traverses the TSP tour \mathcal{T} on centers of all non-trivial clusters of Z_l , and serves the demands in each cluster (as above) when it is visited. The resulting solution has length at most $d(\mathcal{T}) + O(1) \sum_{S \in Z_l} \text{LB}(S) \leq O(1) \cdot \text{LB}$, by Claim 19. Moreover, each object $i \in H_l$ spends at most $O(1) \cdot d(s_i, t_i)$ time in the vehicle.

Finally, concatenating the tours serving each H_l (for $l = 1, \dots, p$), and then the tours serving each G_j (for $j = 1, \dots, \lceil \log_2 \Delta \rceil$), we obtain the 1-preemptive tour claimed in Theorem 15.

References

- [1] I. Abraham, C. Gavoille, D. Malkhi, and U. Wieder. Strong-diameter decompositions of minor free graphs. In *SPAA '07: Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures*, 2007.
- [2] S. Antonakopoulos, C. Chekuri, B. Shepherd, and L. Zhang. Buy-at-bulk network design with protection. In *FOCS '07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 634–644, 2007.
- [3] B. S. Baker. Approximation Algorithms for NP-Complete Problems on Planar Graphs. *Journal of ACM*, 41:153–180, 1994.
- [4] C. Busch, R. LaFortune, and S. Tirthapura. Improved sparse covers for graphs excluding a fixed minor. In *PODC*, pages 61–70, 2007.
- [5] I.-M. Chao. A tabu search method for the truck and trailer routing problem. *Computer & Operations Research*, 29:469–488, 2002.
- [6] M. Charikar and A. Karagiozova. On non-uniform multicommodity buy-at-bulk network design. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 176–182, 2005.
- [7] M. Charikar, S. Khuller, and B. Raghavachari. Algorithms for capacitated vehicle routing. *SIAM J. Comput.*, 31(3):665–682, 2001.
- [8] M. Charikar and B. Raghavachari. The Finite Capacity Dial-A-Ride Problem. In *FOCS*, pages 458–467, 1998.
- [9] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for non-uniform buy-at-bulk network design. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 677–686. IEEE Computer Society, 2006.
- [10] C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for node-weighted buy-at-bulk network design. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1265–1274. Society for Industrial and Applied Mathematics, 2007.
- [11] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. *GSIA, CMU-Report 388*, 1977.
- [12] J.-F. Cordeau and G. Laporte. The Dial-a-Ride Problem (DARP): Variants, modeling issues and algorithms. *4OR: A Quarterly Journal of Operations Research*, 1(2), 2003.

- [13] W. E. de Paepe, J. K. Lenstra, J. Sgall, R. A. Sitters, and L. Stougie. Computer-Aided Complexity Classification of Dial-a-Ride Problems . *Inform Journal on Computing*, 16(2):120–132, 2004.
- [14] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Min-max tree covers of graphs . *Operations Research Letters*, 32(4):309–315, 2004.
- [15] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.
- [16] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. *SIAM Journal on Computing*, 7(2):178–193, 1978.
- [17] I. L. Gørtz. Hardness of Preemptive Finite Capacity Dial-a-Ride. In *APPROX-RANDOM*, pages 200–211, 2006.
- [18] A. Gupta, M. T. Hajiaghayi, V. Nagarajan, and R. Ravi. Dial a ride from k -forest. *ACM Transactions on Algorithms*, 6(2), 2010.
- [19] M. Haimovich and A. H. G. R. Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research*, 10:527–542, 1985.
- [20] D. Hochbaum and W. Maass. Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI. *Journal of ACM*, 32:130–136, 1985.
- [21] S. Khuller, B. Raghavachari, and N. E. Young. Balancing Minimum Spanning Trees and Shortest-Path Trees. *Algorithmica*, 14(4):305–321, 1995.
- [22] P. Klein, S. A. Plotkin, and S. Rao. Excluded minors, network decomposition, and multicommodity flow. In *STOC*, pages 682–690, 1993.
- [23] F. Lazebnik, V. Ustimenko, and A. Woldar. A New Series of Dense Graphs of High Girth. *Bulletin of the AMS*, 32(1):73–79, 1995.
- [24] S. Mitrović-Minić and G. Laporte. The Pickup and Delivery Problem with Time Windows and Transshipment. *Information Systems and Operational Research*, 44:217–227, 2006.
- [25] C. Mues and S. Pickl. Transshipment and time windows in vehicle routing. In *8th Int. Symp. on Parallel Architectures, Algorithms and Networks*, pages 113–119, 2005.
- [26] Y. Nakao and H. Nagamochi. Worst case analysis for pickup and delivery problems with transfer. *IEICE Trans. on Fund. of Electronics, Comm. and Computer Sci.*, E91-A(9), 2008.
- [27] M. Savelsbergh and M. Sol. The general pickup and delivery problem. *Transportation Science*, 29:17–29, 1995.
- [28] S. Scheuerer. A tabu search heuristic for the truck and trailer routing problem. *Computer & Operations Research*, 33:894–909, 2006.